Department copy

EXPERT SYSTEMS AND STATISTICAL EXPERTISE

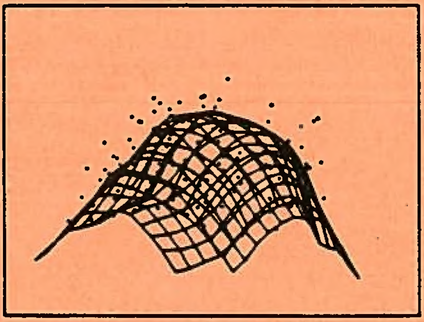
Part I: Statistical Expert Systems

Naomi S. Altman

Technical Report No. 17

May 1985

Laboratory for Computational Statistics



Department of Statistics
Stanford University

Expert Systems and Statistical Expertise
Part I: Statistical Expert Systems

Naomi S. Altman

Dept. of Statistics and Computation Research Group

Stanford Linear Accelerator Center

Stanford University

Abstract

An expert system is a computer program that performs at the level of a human expert in a complex but narrow field. Two types of expert systems which involve statistical expertise are statistical consulting programs and programs which find patterns in databases. Consulting programs can now be built quickly using programming tools. Twosamp, a consultant for advising on the analysis of univariate two-sample problems, was constructed in less than a week using one such tool, EMYCIN. Systems like Twosamp cannot model the initial stages of a consulting session, but can provide expert assistance once a class of models has been selected. Consultants and pattern-finders can assist statisticians in building models, generating hypotheses, and maintaining complex databases. As well, they can serve as laboratories for experimentation with statistical strategies.

Expert systems also use statistical expertise. Most systems include mechanisms for reasoning under uncertainty. Methods under investigation include fuzzy logic, Dempster-Shafer theory, Bayesian analysis and various ad hoc methods. Learning systems use statistics to infer inductive rules. As well, statistical reasoning will be used to evaluate the performance of expert systems.

This work supported by an Office of Naval Research Contract N00014-83-K-0472.

Chapter 1

Introduction

An expert system is a computer program that performs at the level of a human expert in a complex but narrow field. To qualify as an expert system, the program must handle problems in a domain in which expertise is an "art", that is, no algorithmic solution to problems exist, but rather, expertise comes from experience and heuristic reasoning. For example, fitting a curve through a cloud of data by nonparametric smoothing does not qualify as expert behavior – fitting is described by a well-defined algorithm. Choosing the most appropriate smoothing technique is expert behavior – it requires heuristic knowledge about what properties of the data are displayed by each technique, and which are important for the data set at hand. Recognizing that the smooth comes from a given parametric family is also expert behavior. Although in principle, this could be done by searching some set of possible curves, this set is simply too large to search exhaustively without some guiding heuristics.

Part I of this report deals with expert systems whose domains of expertise include some statistics. Part II deals with the use of statistics in building and evaluating expert systems.

Two types of expert systems which involve statistical reasoning are advice-giving programs and pattern-finding programs. Advice-giving systems have encoded within them rules for decision making in some domain, and have the goal of solving or pointing to solutions of a selected set of problems, for example, of diagnosing a disease, or determining an appropriate statistical analysis. Pattern-finding systems search through a database for interesting facts, for example, side-effects of drug use, or rules for reaching certain conclusions.

Advice-giving expert systems involve statistical reasoning if they give statistical advice. (Many advice-giving programs also have devices to handle uncertainty in the data or vagueness in the rules, but I do not include this aspect in my definition of statistical reasoning.) Currently, advice-giving systems have been written in two modes. The consultant attempts to use its rules to solve the problem itself, and turns to the human expert only if it lacks either the data or the rules to continue. Because the program takes the lead in solving the problem, most consultant programs include explanation systems that allow the human expert to query the program's reasoning. Examples of consultant systems are MYCIN (Buchanan and Shortliffe, 1984), which provides medical diagnosis and prescribes drugs for bacterial infections and REX (Pregibon and Gale, 1982) which does an expert job of least-squares linear regression.

The assistant aids the human consultant. The assistant may perform any number of duties, such as keeping track of the data and techniques available to the human consultant, keeping track of what has been or should be done, or performing tasks requested by the consultant. For example, SACON (Buchanan and Shortliffe, 1984), assists an engineer in running a complex computer package, MARC, which analyzes structural characteristics of materials, such as stress. REFEREE (Buchanan, Brown et al, 1984) assists an editor in assessing the quality of a paper describing a clinical trial. PA (Waters, 1982), the Programmer's Assistant, constructs computer programs by interpreting commands like "iterate", and "maximize".

Pattern-finding systems search a database to discover relationships. These systems use statistical techniques such as correlations and discriminants as well as knowledge about their domain. For example, the RX program, (Blum, 1982), searches a medical database for possible causal relationships, where causation is defined by lagged correlation. The program then uses medical knowledge to rule out common causes and clearly spurious correlations. Finally, both statistical and medical knowledge are used to test the correlation, after controlling for other associated variables. The Odysseus system (Wilkins et al., 1985), attempts to create diagnostic variables by use of partitioning techniques in a medical database. No causality is assumed. The program attempts to find the symptoms associated with disease states, and thus to add rules to a diagnostic consultant. The program requires medical knowledge, statistical knowledge, and knowledge about the structure of plausible rules.

A statistical consulting system is simply a better package. Existing packages compute accurately. However, they are highly sensitive to minor syntax errors, and do not provide much guidance in the use of statistical techniques beyond instructions for entering the data and running the program. They do not guard against even the most blatant or readily detected misuse of techniques, and often produce reams of output without much interpretive assistance.

Typically, expert systems have more flexible language handling abilities. This means the user need not be annoyed so often by simple syntax and spelling errors.

More important, expert systems have heuristic knowledge which can guide the analysis and assist the interpretation of the results. For example, most linear regression packages note computational problems, such as singularity of the design matrix, since the numerical algorithms will fail. Few such packages note even commonly occurring and readily detected problems such as integer-valued response variables or very small degrees of freedom for error, since these do not interfere with the numerical algorithms. An expert system could have encoded within it guidelines for the appropriate use of various statistical techniques. An expert regression package, for example, would draw the user's attention to violations of the normality assumptions and warn of problems like overfitting. At a higher level, an expert statistics package could advise a user who wants to know the "relationship" between two variables, that a particular technique is suitable.

Many packages do provide diagnostics and other information automatically. However, these are often buried within the output. The user with little statistical expertise may not understand the use of these statistics or may not how to proceed if she does recognize a problem. (Some well-known packages produce statistics for the regression problem that most expert statisticians are not familiar with.) An expert system could examine the diagnostics and draw the user's attention to those with significant implications for the analysis, such as the existence of a highly influential point. The essential difference between a package and an expert system is in the phrase "draw the user's attention". The expert system computes, but does not print, all the relevant statistics. Like the human expert, the expert system interprets these results, and then advises the user on the appropriate course of action. Current packages rely on the user to recognise the important diagnostics, and to know how to proceed.

Of course, a statistical expert system need not have a numerical component. For example, an expert system knowledgeable about experimental design could assist a client in planning a study. The important points are that the expert system goes beyond a textbook by dealing with the problem presented by the user, not with the abstract case, and goes beyond the package by interpreting aspects of the analysis and bringing the important features to the attention of the user.

An expert system does not have the abilities of a human expert. In narrow range of expertise, it may well out-perform a human expert due to the superior data retrieval and computational abilities of the computer. However, the human expert can perform in a far wider range of problems. She has "world knowledge", that is a broad range of knowledge that lies outside her field of expertise. And she learns readily both from experience and from other experts. To date, coding world knowledge into a program is not possible. And, although machine learning is an area of current research, programs do not currently have the understanding and insight of humans. The expert system has only domain knowledge, and has only a limited ability to extend that knowledge.

Chapter 2

Building an Expert System

The first expert systems were coded by programmers working in areas in which they had expertise, such as games. However, in fields of specialized knowledge, such as chemistry and medicine, the programming expertise and domain expertise are generally held by different individuals. Furthermore, expertise in these areas is often implicit, embodied in rules of thumb, and transferred, not only by textbook training, but also by "hands-on" experience. Communication between the domain expert and the programming expert is a major problem.

One design strategy used to alleviate this problem was to separate the "domain knowledge" of the system from its "control knowledge" or "inference engine". The domain knowledge consists of the facts and heuristics known to the expert. For example, the knowledge that a t-test is a test of location is knowledge in the statistical domain. This knowledge is usually encoded as a series of rules. For example:

> **"If the data are Gaussian,**
> **and the test is a test of location,**
> **use a t-test."**

The "control knowledge" is the problem solving strategy used by the program. For example, in symbolic integration, many equivalent answers are possible, and only one is needed, but in medical diagnosis it is necessary to keep track of all possible diagnoses of the patient.

There have been two major consequences of this separation of domain knowledge and inference engine. Firstly, the domain expert, with only a cursory knowledge of the inference engine, can add to and change the facts and heuristics in the knowledge base. The second is that the same inference engine can be used within different domains of expertise if the problem solving strategies can be couched in similar terms. Only a new knowledge base must be constructed.

Knowledge acquisition systems such as TEREISIS (Buchanan and Shortliffe, 1984), and KAS (Hayes-Roth et al., 1983), are programs designed to assist the domain expert in building a knowledge base. Inference engines and knowledge acquisition systems have been assembled into tools such as EMYCIN (Van Melle et al., 1984), and KAS which can, in principal, be used directly by the domain expert to build an expert system. The expert requires about the same level of programming skills required to use a statistical package. Whereas building an expert system from scratch is a matter of years, systems can be built in a matter of months using these tools. For example, MYCIN (Buchanan and Shortliffe, 1984), a medical diagnosis system took 4 years to build, whereas my very simple system, Twosamp, was built in less than a week using EMYCIN, the expert system building tool derived from MYCIN, and SACON (Buchanan and Shortliffe, 1984), was built in about 4 months using the same tool. MYCIN has about 500 rules of domain knowledge, Twosamp has 38, and SACON has 170.

There are several types of inference engines now in use in various expert systems. EMYCIN assumes that all domain knowledge is encoded in production rules and that inference is done by backward chaining from the goal. A production rule is an "if-then" statement, such as

"If the problem is to detect a difference in location,
and if the data are paired,
and if the differences are Gaussian,
then the appropriate procedure is paired t-test".

Backward chaining means that the inference engine starts from the goal, and seeks all information needed to attain that goal. In the example above, the goal is to determine the appropriate procedure to use. In order to determine this, it is necessary to determine the type of problem, if the data are paired, and if the differences are Gaussian; these become the new goals of the system. EMYCIN allows for multiple goals, and multiple conclusions.

EMYCIN also allows for uncertainty in the user's knowledge of the parameters, and in the conclusions of the rules. These are dealt with using "certainty" factors, an ad hoc method, which is explained in part II of this report. I did not use certainty factors in building Twosamp.

Chapter 3

Building Twosamp

Twosamp is a statistical assistant system, for advising on the appropriate analysis in the univariate two-sample location problem. The system was designed as a student project in a one semester reading course. It took about 8 hours to design (on paper) a decision tree for this problem, and another 30 hours to learn EMYCIN and enter the rules. The primary path of the decision tree is displayed in Figure 1. A small set of rules, as they are interpreted by EMYCIN, is displayed in Figure 2. Part of a Twosamp run is displayed in Figure 3.

Which primary node of the tree is selected is determined by three variables, the data type, the design, and the model. Five data types are considered, dichotomous, categorical, count, continuous and ranked. (If the study involves survival or lifetime data the client is advised to see a statistician. This was done only to keep the problem small, since selecting an appropriate technique in survival analysis involves another large decision subtree.) Designs considered were matched and unmatched data. The two models depend on whether the grouping is the independent or dependent variable. If the grouping is the independent variable, the problem is to determine if there is a difference in distribution of the measured variable between the two groups. If the grouping is the dependent variable, the problem is to determine how group membership depends on the measured variable (that is, logistic regression or a related model.)

The ordering of the primary nodes is not arbitrary. It is forced by the logic of EMYCIN. EMYCIN allows only backwards chaining and conclusions reached by EMYCIN are irrevocable. For example, Twosamp asks questions about the differences between pairs, only if it has determined that a paired t-test may be an appropriate technique, that is, only under the node

CONTINUOUS=true
MODEL=difference-in-mean
N-MATCHED=1

Figure 1

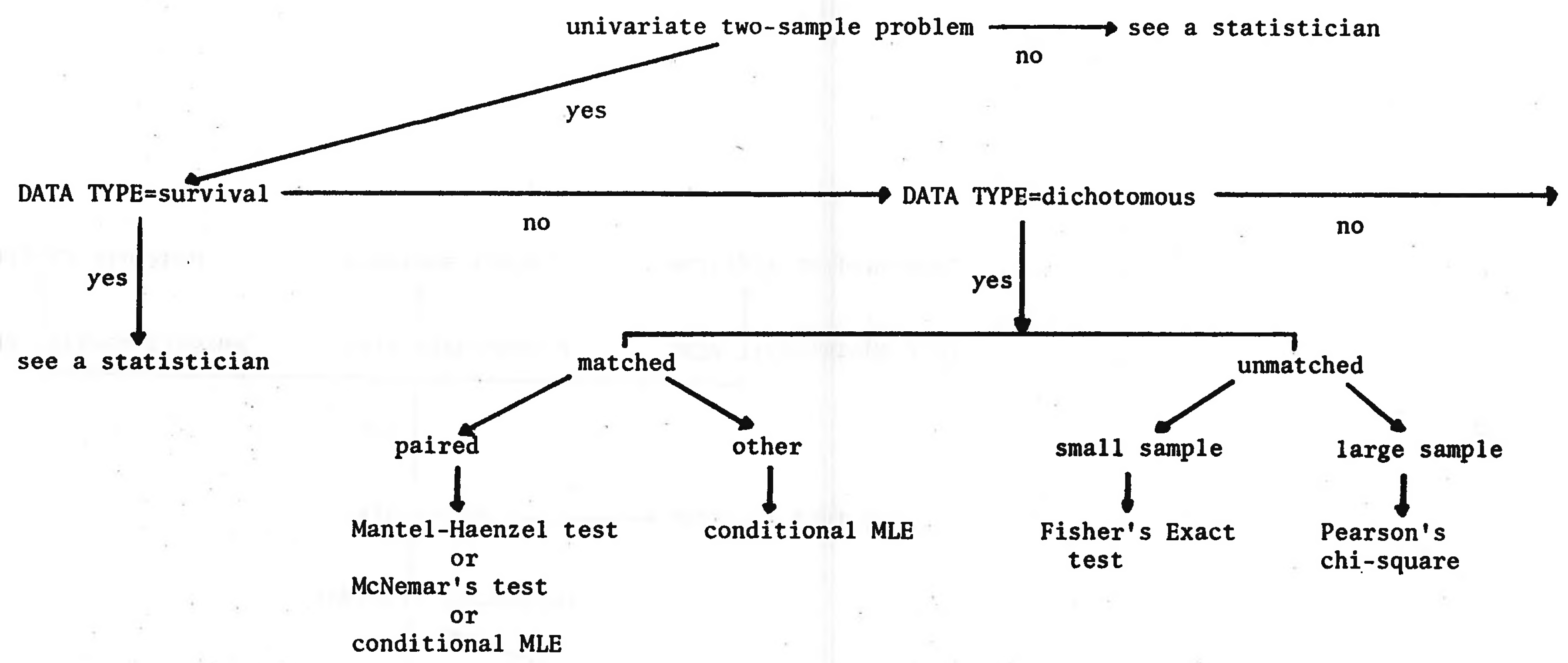

univariate two-sample problem
no
see a statistician
yes
DATA TYPE=survival
yes
see a statistician
no
DATA TYPE=dichotomous
no
yes
matched
unmatched
paired
other
small sample
large sample
Mantel-Haenzel test
or
McNemar's test
or
conditional MLE
conditional MLE
Fisher's Exact test
Pearson's chi-square

Figure 1 cont.

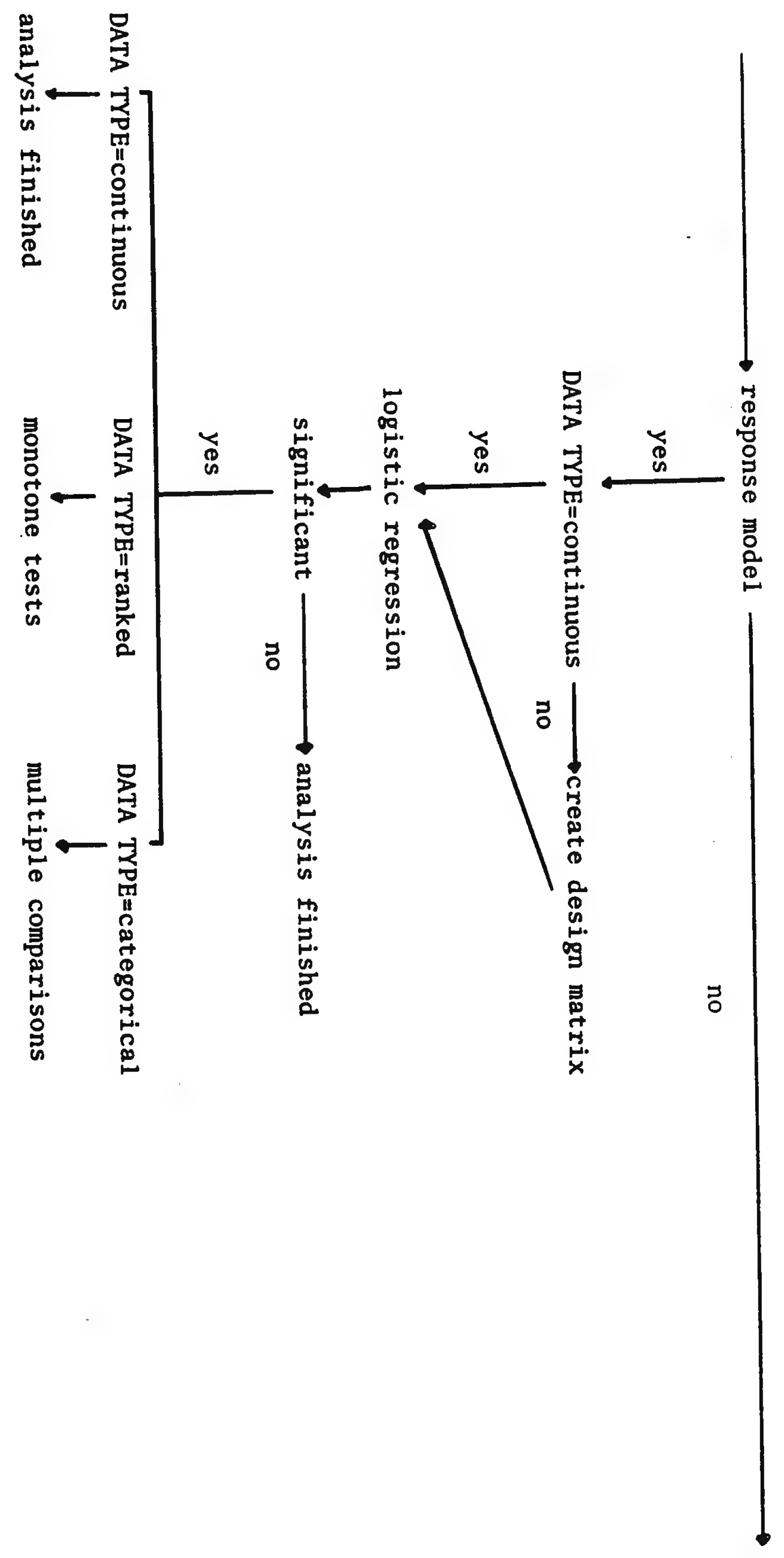


Figure 1 cont.

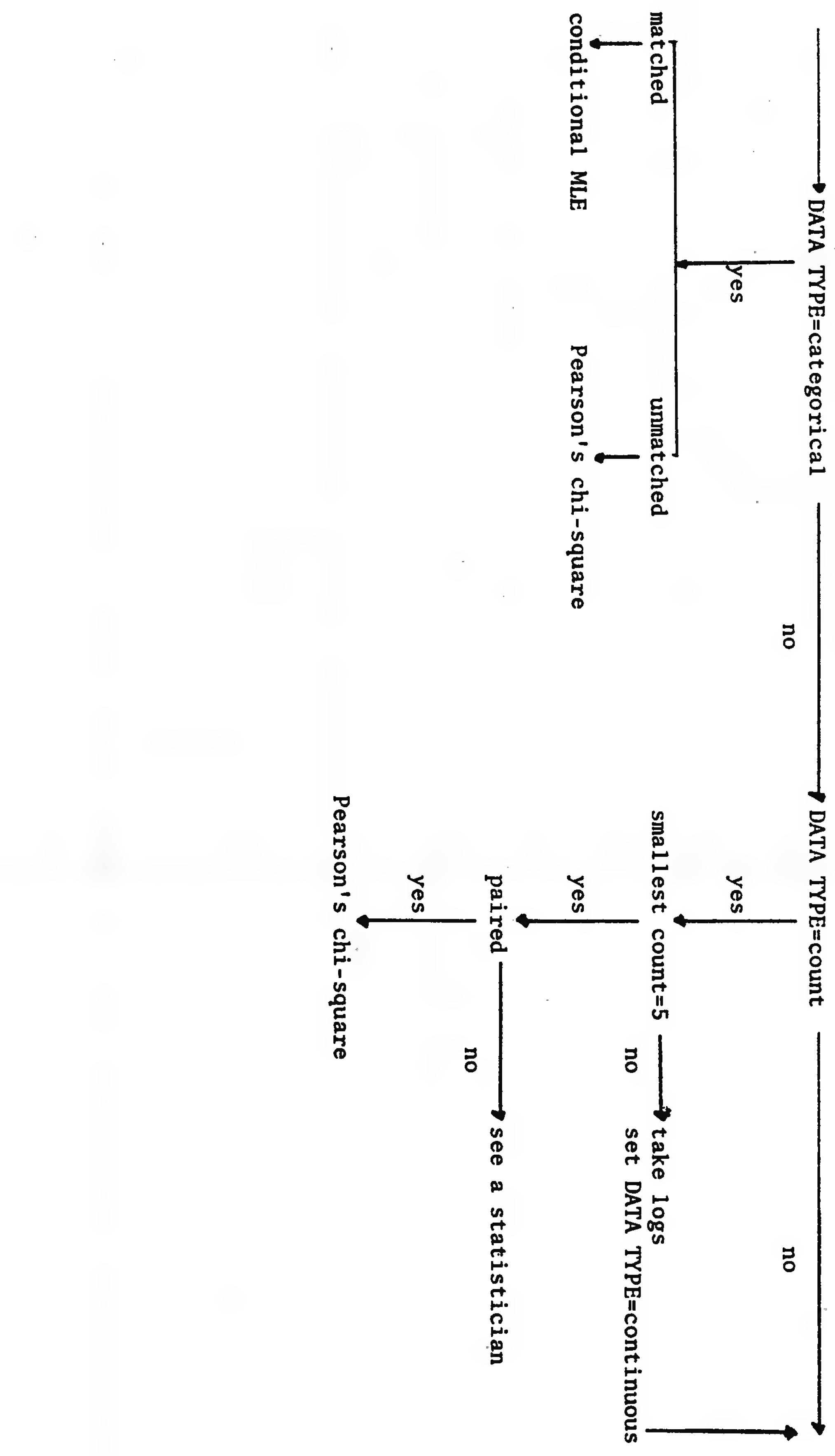


Figure 1 cont.

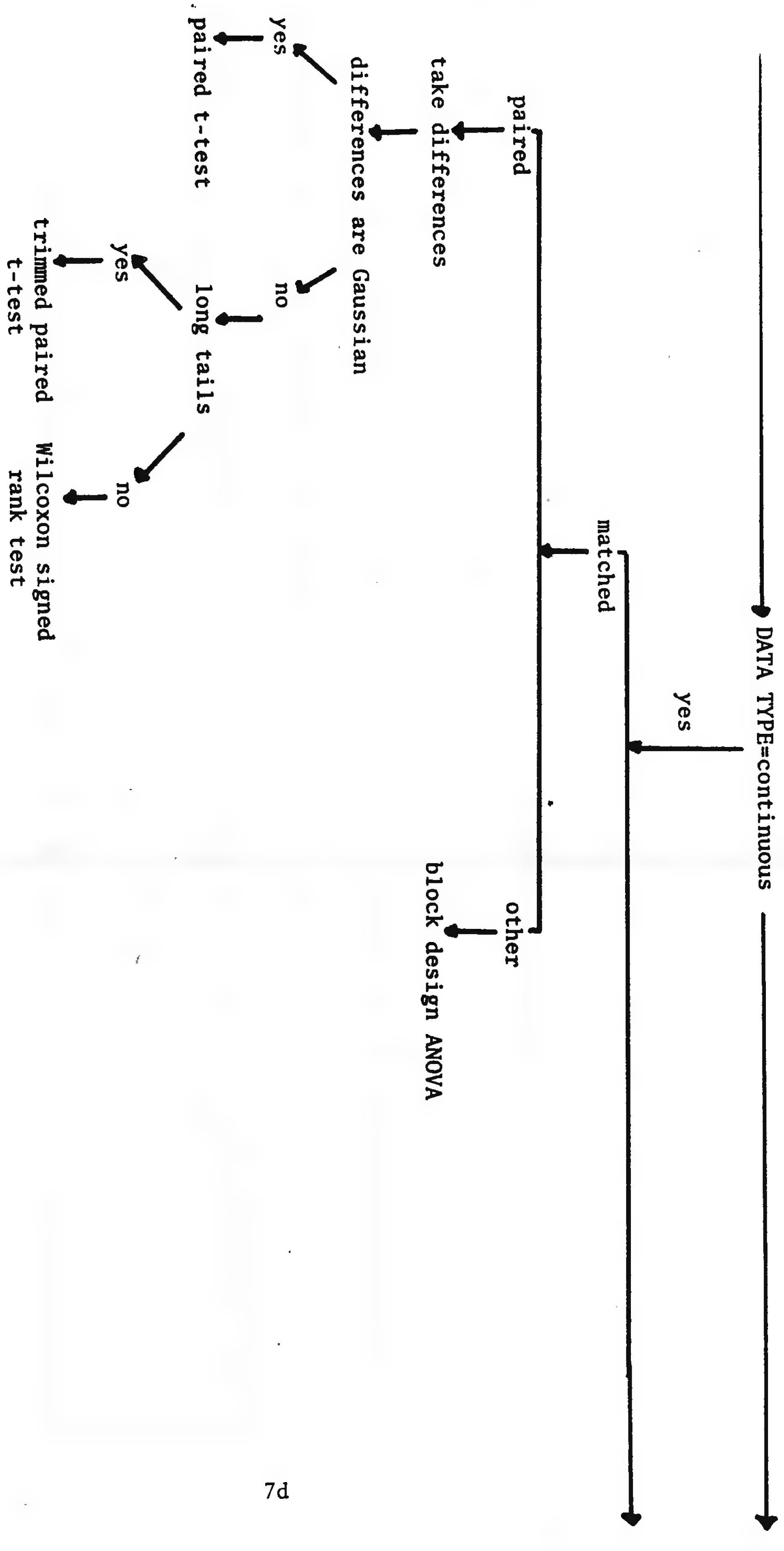


Figure 1 cont.

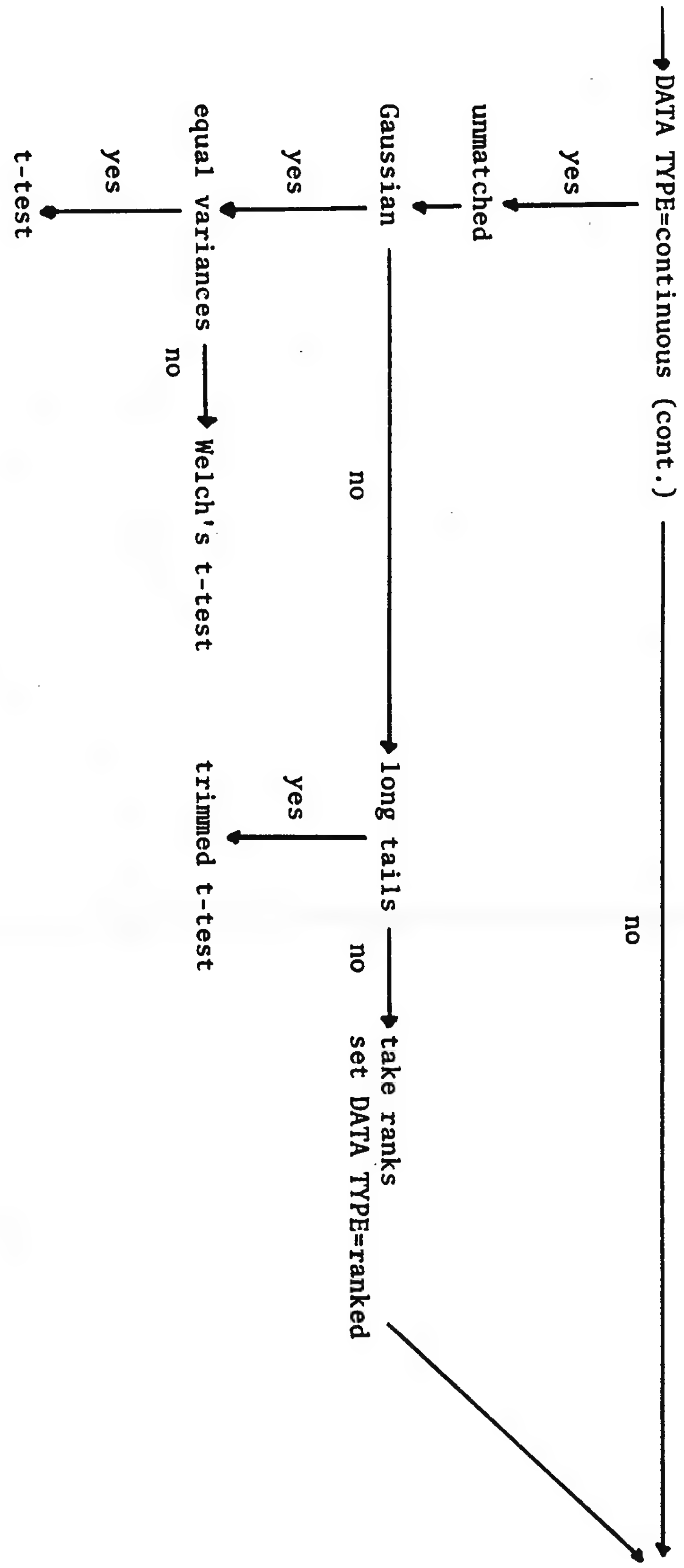
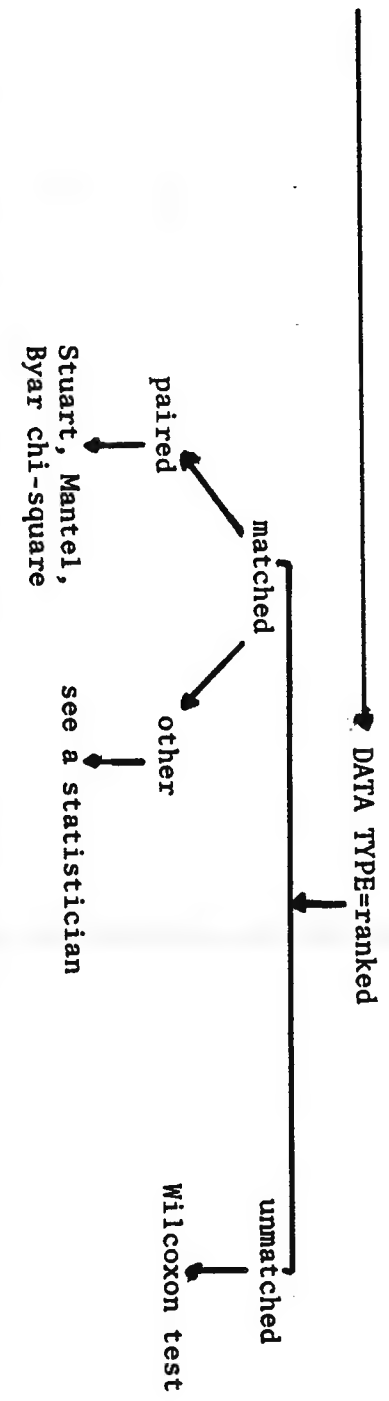


Figure 1 cont.



In statistics, attributes of the data may change as the analysis proceeds. For example, after taking logarithms, data that were originally counts may be treated as continuous, or data that were skewed may be considered Gaussian. Since EMYCIN does not allow re-evaluation of problem parameters, decision nodes must appear in an appropriate order, and dummy variables are occasionally needed.

Rules are entered using EMYCIN's knowledge acquisition routine. Rules may be entered in a simplified "natural language", more tersely in "mathematical language", or in Lisp. Examples of all three types of entry are displayed in Rule 007 in Figure 2. EMYCIN prompts for definitions of any new parameters encountered, and translates the rules into all three modes, using simple templates. The templates for converting from mathematical language to Lisp are provided by EMYCIN. The templates for converting to natural language are provided by the human expert. For example, the template for the variable name "ADVICE" is "one of the first things you should do" and the template for its value "DESIGN" is "create a design matrix". EMYCIN uses grammatical rules to create English-like sentences from the templates. Spelling correction is done automatically for known parameters, and other corrections may be done via the line editor. EMYCIN queries the user if rules give conflicting advice.

Rules fire when their premises are true. The first rule to be attempted is the goal rule. The action of the goal rule is to end the consulting session. The goal rule of Twosamp is Rule 038, displayed in Figure 2. Since the program attempts to resolve the truth of the premises of the rule, it will first attempt to determine if the problem is a univariate two-sample problem, and if it is, to find the appropriate analysis. The attempt to resolve the premises of the goal rule controls the firing of the other rules.

A consulting session is recorded in Figure 3. Notice that EMYCIN provides spelling correction and an explanation facility (the "WHY" command.) Due to an EMYCIN bug, the program has not attempted to resolve the first premise of the goal command, that is, has not first checked to ensure that this is a univariate two-sample problem. The program can give two levels of advice. The "solution" is a single piece of advice, that signals the goal rule to finish the session. If the solution needs to be modified, for example, by reminding the user to create a design matrix, another variable, "initial advice" is also printed. This problem displays a common piece of advice "See a statistician." This advice is basically a place-holder, indicating problems that do not have standard solutions, or nodes in the decision tree such as "DATA-TYPE=survival" which are not currently handled by the program, but which could be added later.

My goal was to design a system that handled a consulting session with a user who was somewhat conversant with statistical language, but not knowledgeable in the full range of tools available to handle the two sample problem. The proliferation of rules was limited by referring the client to a human statistician if the problem had no standard solution, or more properly belongs to the realm of survival analysis. As a result, although the problem domain is small, 38 rules were needed.

Figure 2

```
RULE007
-------
[This rule is tried in order to find out about the first thing you
    should do or the solution]

  If:  1) The problem is prediction, and
       2) The type of data is the data are unordered categories
  Then:  1) It is definite (1.0) that the following is one of the
            first thing you should do: create a design matrix,
         2) It is definite (1.0) that the following is one of the
            first thing you should do: use logistic regression, and
         3) It is definite (1.0) that the following is the solution:
            use multiple comparisons if the logistic regression was
            significant

  Premise: ($AND (PREDICTION=PREDICTION) (DATA-TYPE=CATEGORICAL))
  Action:  ($AND (ADVICE=DESIGN) (ADVICE=LOGISTIC) (FINAL-ADVICE=MULTIPLE))

  PREMISE:  ($AND (SAME CNTXT PREDICTION PREDICTION)
                  (SAME CNTXT DATA-TYPE CATEGORICAL))
  ACTION:  (DO-ALL (CONCLUDETEXT CNTXT ADVICE (TEXT DESIGN)
                                 TALLY 1000)
                   (CONCLUDETEXT CNTXT ADVICE (TEXT LOGISTIC)
                                 TALLY 1000)
                   (CONCLUDETEXT CNTXT FINAL-ADVICE (TEXT MULTIPLE)
                                 TALLY 1000))


RULE038
-------
[This rule is tried in order to find out about whether the analysis
    is finished]

  If:  1) This is not a two-sample one variable problem, or
       2) The solution is known
  Then:  1) It is definite (1.0) that the analysis is finished,
         2) Display the first thing you should do, and
         3) Display the solution

  Premise: ($AND ($OR (INAPPROPRIATE) (KNOWN CNTXT FINAL-ADVICE))
  Action:  (DO-ALL (FINISHED) (PRINTCONCLUSIONS ADVICE)
            (PRINTCONCLUSIONS FINAL-ADVICE))

  PREMISE:  ($AND ($OR (SAME CNTXT INAPPROPRIATE)
                       (KNOWN CNTXT FINAL-ADVICE)))
  ACTION:  (DO-ALL (CONCLUDE CNTXT FINISHED YES TALLY 1000)
                   (PRINTCONCLUSIONS CNTXT ADVICE T)
                   (PRINTCONCLUSIONS CNTXT FINAL-ADVICE T))
```

Figure 3a

```
   --------PROBLEM-1--------
   1) Are the data dichotomous categorical, ordered, ranks, counts, or
      continuous?
   ** RANKS
   2) Do you want to determine if there is a significant difference in the
      distribution in the two groups (difference-in-mean) or predict group
      membership?
   ** DIFF
=DIFFERENCE_IN_MEAN
   3) Are the data in the first group matched in any way to the data in the
      second group?
   ** Y

The solution is as follows:
        See a statistician.
```

Figure 3b

```
---------PROBLEM-2---------
1) Are the data dichotomous categorical, ordered, ranks, counts, or
   continuous?
** COUNTS
2) Do you want to determine if there is a significant difference in the
   distribution in the two groups (difference_in_mean) or predict group --
   membership?
** DIFFERENCE_IN_MEAN
3) What is the smallest count in the data?
** WHY

[i.e. WHY is it important to determine the smallest count?]

[1.0] This will aid in determining whether the data are continuous.
     It has already been established that
        [1.1] the type of data is the data are counts
     Therefore, if
        [1.2] the smallest count is greater than or equal to 5
     then
        it is definite (1.0) that the data are continuous
     [ALSO:
        it is definite (1.0) that the data have been tested to determine if
           they are Gaussian
        it is definite (1.0) that the data are Gaussian
        Take the logarithm of the data.]
     [RULE013]

   [back to question 3...]
   ** 7

Take the logarithm of the data.

   4) Are the data in the first group matched in any way to the data in the
      second group?
   ** Y
   5) How many experimental units in group two are matched to a single unit
      in group one?
   ** 1

Take the differences between pairs. From this point onwards we refer only to
the differences.

The solution is as follows:
        Do a one-sample t-test on the differences.
```

Twosamp does not access statistical package. That is, it gives advice about which technique to use, but does not do any analysis for the client. This makes Twosamp very different from programs like REX in which the consultant carries out the entire analysis. Twosamp does not have much knowledge about the data and must always query the client. In this mode, Twosamp is simply a summary of its decision tree.

Even this small system could be quite useful, however, if attached to a statistical package. This could be done without major revision to the current knowledge base, although new rules would have to be added to interpret the output of the package. Passing the information back and forth from the package to the system would be a tedious programming task. Current systems which access statistical packages, such as RX (Blum, 1982), and REX (Pregibon and Gale, 1982), handle this by allowing some rules to create calls to the associated package and then scanning the output for relevant values.

Although Twosamp does an adequate job of the actual data analysis task, it does not model a consulting session. A statistical consultant asks general questions of the type "Where does this data come from?" "How did you collect it?" and so on. Twosamp has no general world knowledge, nor does it understand natural language well. **Twosamp enters the consulting session after the human consultant, through dialogue with the client, has narrowed the problem to selecting within a small set of possible models (techniques).**

Chapter 4

The Consultant versus the Assistant

Since the idea of building a statistical consultant expert system was suggested by J. Chambers at the 1981 Interface of Computer Science and Statistics Interface (Chambers, 1981), there has been considerable debate in the statistical community about the role of the program as a consultant. While some statisticians feel that a consultant system would assist users who currently use statistical packages without consulting a statistical expert, others worry about putting yet a more powerful tool in the hands of the untrained user.

The argument against the consultant program, is that the consultant, due to lack of real world knowledge, and of natural language recognition, cannot pursue vague questions of the type "Where did this data come from?" or "What are you trying to find out?" These are the questions that are often the crux of a session with a human consultant. That is, the task of the consultant is not just to suggest and carry out a statistical analysis of the client's data, but to clarify the premises, actions and goals of the client.

It is useful, in thinking about this question to note that experts in other fields are also uncomfortable about using consulting systems. Comments like "The computer cannot understand how sick this patient really is" (how complex this data set really is) or "Every patient (data set) is unique" were made by assessors of MYCIN (Buchanan and Shortliffe, 1984). Yet in blind assessments, the MYCIN compared well with human experts. Of course, these assessments are done only using cases for which the use of the program is already indicated.

There are two major differences between consultation systems for statistics and for those in other fields. In the first place, in a field such as chemistry or medicine, the computer consultant has knowledge about the problem domain. The statistical consultant, on the other hand, has knowledge about the statistics domain, not about the problem domain. Secondly, medical consulting systems are designed to be used by doctors, not by their patients. The consulting system provides expert advice, when used by a general practitioner who has already recognized a class of possible disease states. However, given the current use of statistics packages by clients with only cursory knowledge of statistics, the builders of statistical consultation systems must realistically expect that these systems will be used by clients with little statistical expertise.

Figure 4 is a simple model of a statistical consulting session. The initial stages of the session consist of dialogue between the consultant and the client. The purposes of this dialogue are to reach a common vocabulary for discussion of the problem, to determine the characteristics of the available data, to determine the goals of the client, and, finally, to determine a class of models (techniques) which may be appropriate. The second stage of the session is the analysis loop. At this stage, the data are examined in detail. Summaries such as graphic displays, tables, and approximating equations may be used. Interesting features of the data are noted. Statistical tests are performed. The analysis stage is iterative. A careful analysis usually requires several cycles through the analysis loop. In each cycle, the data analyst, who may be the statistician or the client, examines interesting features, such as outliers, and associations between variables, and adjusts the model. There may also be periodic returns to the dialogue step. The final step is writing the report on the statistical aspects of the problem.

Statistical consulting systems which can handle the dialogue stage of the consulting session could be built for very limited problem domains. For example, a system which could design and analyze clinical trials in a small medical domain may be possible, since a few hundred rules could probably handle most of necessary knowledge. However, the same system would not be able to handle similar statistical techniques in another domain, because the vocabulary of the problem and the accompanying domain knowledge would differ. It is not currently feasible to build statistical consulting systems which can handle the dialogue stage of the consulting session for a general consultant.

It is more realistic to suppose that the statistical consulting system will enter the consulting session at the second stage, during the analysis loop. With access to a statistical package, the system can handle the iterations of the analysis loop, once a class of models have been identified. The system can also keep track of what it has done, and write a report.

In this sense, statistical consulting systems have the same limitations as medical consulting systems. Medical systems do not model the role of the general practitioner. Such a system would require too much world knowledge. Medical computer consultants work well in a very limited domain. The patient is already known to have a disease in a given (small) class (for example, meningitis) and the remaining task is simply to narrow the diagnosis further in order to prescribe appropriate treatment. Similarly, the computer statistical consultant can do an expert job of the analysis

Figure 4

$$\begin{tikzcd}
\text{client} \arrow[r] & \text{DIALOGUE} \arrow[d] & \text{statistician} \arrow[l] \\
& \begin{array}{c}\text{CLASS OF MODELS}\\ \text{(TECHNIQUES)}\end{array} \arrow[d] & \\
& \text{ANALYSIS LOOP} \arrow[d] & \\
& \text{compute} \arrow[d, bend right] & \\
& \text{interpret} \arrow[u, bend right] \arrow[d] & \\
& \text{REPORT} &
\end{tikzcd}$$

once a class of techniques has been selected by the human consultant.

Many of the arguments between opponents and proponents of the use of expert systems for statistics turn on this distinction. The opponents have focused on the dialogue stage of the consulting session. At the current state of the art, expert consulting systems do not handle this well. The proponents of expert systems have focused on providing expert analysis once the human expert has formulated a class of models. Programs such as REX (Pregibon and Gale, 1982), demonstrate that this is a feasible task.

Statistical consultation systems can be useful for the applied statistician. Even a trained statistician cannot have expertise in all the techniques she knows. It is far easier to identify a class of models which suit the problem, than to do an expert job of finding a specific model to fit a particular data set. Applied statisticians must often provide consulting support over a wide range of problem areas. Expert systems could provide them with much needed support, especially for new techniques.

Expert systems will be useful pedagogically. Just as the development of statistical packages has freed us from teaching students the details of the numerical algorithms, the development of expert systems will free us from teaching non-technical students the technical details of a good analysis. What must be taught are the uses and interpretation of statistical techniques.

For the time being, at least, statistical consulting programs, like statistical packages, are an asset for statistician, but carry the potential for abuse by the non-statistician. Statistical packages reduced programming errors but did not prevent users from running an inappropriate analysis. Statistical expert systems, too, cannot prevent users from doing an inappropriate analysis, but they do ensure that the selected analysis is done well. Of course, if an inappropriate analysis is selected, the quality of the analysis is irrelevant. However, this is a problem which already exists. At least for that level of client with some statistical knowledge, the consulting system can provide guidance, and can reduce some of the manual labor of performing the analysis loop.

Chapter 5

Statistical Expertise and Expert Statisticians

Building an expert consulting system is like writing a textbook or a manual for a statistical package. It codifies and explicates existing statistical techniques and strategies. New techniques arise only minimally, when numerical summaries are required for visual displays, or gaps are discovered in current practise.

The development of tools like EMYCIN may, however, change the manner in which statistical software is developed. Commercial packages will undoubtedly begin to distribute expert systems with their routines, and such systems may even replace manuals. (The use of the term "expert system" in this context is unfortunate, since it seems to imply more than such systems will actually deliver.) When expert system building tools like EMYCIN interface more readily with computational programs, it will be practical and attractive for new methods to be released by statistical investigators as expert systems, rather than as statistical algorithms.

Pattern-finding expert systems, such as RX and Odysseus, that try to find new knowledge in a data set have been somewhat neglected by statisticians interested in expert systems. These systems require both domain knowledge and statistical knowledge. They provide interesting laboratories for testing our ideas of the meaning of statistical ideas like causation. Besides, the data sets used in these systems are usually rich, and beg new analysis techniques.

Whatever our feelings as statisticians about the role of expert systems, we cannot afford to ignore them. Expert systems involving statistical expertise will be built, because of the utility to the builders, if not to the potential client. With the explosion of personal computing, writing statistical software has become a lucrative business. Statistical software wrapped in an expert system is that much more marketable. For example, both EMYCIN and BMDP (BMDP, 1983), are now marketed for use on (different) personal computers, and a system which consults on any given BMDP routine could easily be written in a few days. Pattern-finding systems are still at a more primitive stage – they must be built from scratch – but are attractive research topics for students in Artificial Intelligence as well being of obvious interest for institutions handling large databases. The same computer that collects patient records for a medical project could, in its "spare time" search for interesting relationships in the data and generate new hypotheses for research.

Acknowledgements

I would like to thank the following for their assistance and comments: B. Buchanan and D. Wilkins of the Heuristic Programming Project, Stanford University, Dr. B. Brown, Department of Biostatistic, Stanford University and P. Velleman of the Department of Economic and Social Statistics, Cornell University.

References

R. L. Blum, 1982
Discovery, Confirmation, and Incorporation of Causal Relationships from a Large Time-Oriented Clinical Data Base: The RX Project
Computers and Biomedical Research 15.

BMDP Statistical Software, 1983
University of California Press.

B. G. Buchanan and E. H. Shortliffe, 1984
Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project
Reading, Mass., Addison Wesley.

B. G. Buchanan, B. W. Brown, D. E. Feldman, and J. Haggerty, 1984
REFEREE - Research Plan.

J. M. Chambers, 1981
Some Thoughts on Expert Software
in *Computer Science and Statistics: Proceedings of the 13th Symposium on the Interface*, ed. K. W. Heiner, Springer-Verlag.

F. Hayes-Roth, D. A. Waterman, and D. B. Lenat, 1983
Building Expert Systems
Reading, Mass., Addison Wesley.

D. Pregibon and W. A. Gale, 1982
REX: an Expert System for Regression Analysis
in *Computer Science and Statistics: Proceedings of the 14th Symposium on the Interface*, ed. K. W. Heiner, Springer-Verlag.

W. Van Melle, A. C. Scott, J. S. Bennett, and M. Peairs, 1984
The EMYCIN Manual
Heuristic Programming Project of Stanford University Technical Report HPP-81-16.

R. C. Waters, 1982
The Programmer's Apprentice: Knowledge Based Program Editing
IEE Transactions on Software Engineering, Vol SE-8, No. 1, January.

D. C. Wilkins, W. J. Clancey, and B. G. Buchanan, 1985
Learning by Watching: Transfer of Expertise for Classification Expert Systems
submitted to the *Ninth International Joint Conference on Artificial Intelligence.*